Ground Control Station Development for the Dexterous Orbital Servicing System

Paul G. Backes, Greg Tharp, Bruce Bon, and
Samad Hayati
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Linh Phan
SoHaR Incorporated
Beverly Hills, California

Abstract

The development of a ground control station for Earth based control of the Dexterous Orbital Servicing System (110SS) is described. The DOSS flight test program provides a manipulator which is mounted on a multi-purpose experiment support structure (MPESS) in a Space Shuttle payload bay and may execute tasks from the MPESS or while held by the SSRMS manipulator. The DOSS will demonstrate the ability to perform dexterous robotic operations both from an onboard control station and from an Earth-based control station. The initial prototype ground control station for DOSS is described.

1. Introduction

The Dexterous Orbital Servicing System (1)0SS) flight. test program is being developed by NASA to verify the ability to utilize telerobotics for servicing Space Station Alpha [1]. The system will be developed as a class B payload and could be launched in the Fall of 1996. In coordination with the 1)0SS flight test program, NASA initiated the Ground Control Station for DOSS task in fiscal year 1994 to develop a telerobotics control station for commanding space robots from Earth, with focus on the 1.)0SS application. This paper briefly reviews the 1)0SS program and then describes the Ground Control Station development.

The 1)0SS program is being developed by a team of three NASA centers, Johnson Space Center, Langley Research Center and Jet Propulsion Laboratory, and Martin Marrietta Astronautics as expert contractor. JSC provides the project management and is responsible for the formal Orbiter integration process. Martin Marrietta Astronautics (MMA) will develop the majority of the flight systems, including the flight manipulator, flight avionics and the aft flight deck command and display systems. MMA will deliver the integrated payload bay elements to Kennedy Space Center for integration into the Orbiter. Langley Research Center will provide the Hydraulic Manipulator Testbed (HMTB), a hydraulic 1)0SS simulator, for software tests, task panel check-outs, and flight crew training, Jet Propulsion Laboratory will develop an integrated ground control station that will serve as an engineering model of a Payload Operations Control Center (POCC). The ground control station will provide real-time video, graphics and predictive displays, off-line task sequencing and verification, autonomous control command generation, and high rate telemetry feedback and display. The ground control station will be tested via remote operation of the HMTB at LaRC from JPL, and will be delivered to JSC for integration into the flight POCC.

1.1. Mission Objectives

The DOSS has three primary program mission objectives: risk mitigation, demonstration of telerobotics as an operational tool, and technology testbed. Telerobotic maintenance is required for International Space Station Alpha (ISSA) and both ground-based and and Station-based control of external manipulators is planned. Since no other flight tests of telerobotics systems are planned prior to operation on ISSA, the DOSS flight test provides risk mitigation by demonstrating the technologies and finding problems early so that they can be corrected prior to operation on ISSA. Ground control technologies which may be demonstrated include remote control with time delay, scene calibration, safety assurance, and task command sequence generation, verification, and execution. DOSS will verify mechanical interfaces and operational scenarios including Orbital Tool Changeout Mechanism, Orbital Replacement Unit (ORU) and tools interfaces, ORU changeout tasks, alignment, mating, and demating tasks, inspection and verification tasks, and IVA on-orbit control. 1)0SS will demonstrate dexterous telerobotics as a viable operational tool for future Shuttle operations. Telerobotics may provide an alternative to some payload mechanism redundancies, and provide and alternative to astronaut EVA either for se-

lected end-to-end tasks or for EVA equipment setup and stowage to 1('(111 ('(' the amount of EVA astronaut time.

The technical mission objectives support the program mission objectives. The utility of telerobotics for execution of realistic on-orbit operations will be demonstrated. The manipulator design and on-orbit task performance will be analyzed to improve future space telerobotic systems. An aft flight deck workstation will be developed for on-orbit teleoperation with future ability to accept commands from a .groulld-based control station. Comparisons of ground based simulations with flight data will be used to increase the fidelity of ground based simulations.

1.2. Experiment 1 Iardware and Tasks

The 1)0SS flight system hardware components include a payload bay element and an Aft Flight Deck (AFD) Workstation element. The payload bay element includes the manipulator avionics, cameras, task panel, and a Multi-purpose Experiment Support Structure (MPESS). The task panel includes four different protoflight unit Space Station ORUs which will be used for Space Station type ORU tasks. These ORUs represent 85 percent of the ISSA robot-compatible ORUs. The 1)OSS body will have two camera and light subassemblies with pan/tilt capability. The AFD Workstation includes a portable color computer, display software, handcontrollers, and power and data interfaces.

The Ground Control Station (GCS) will be in a Payload Operations Control Center at JSC and will communicate with the flight DOSS system via '1'1)1{SS satellite communications.

The tasks to be performed during the mission fall into five sequential stages: initialization and checkout, fixed base operations and characterization, SRMS-based dynamics evaluation, SRMS-based operations, and stowage and deployment. Initialization and checkout will verify that 1)0SS is working properly and prepare for the subsequent stages. The fixed-base operations and characterizations stage will include tasks where the manipulator base is fixed to the MPESS. Tasks will include ORU targeting, grapple, **removal,** and insertion. In the SltMS-based dynamics evaluation stage, the DOSS manipulator will be picked up by the Space Shuttle Remote Manipulator System. The 1)0SS manipulator will then be commanded to move through a series of autonomous motions to provide data for dynamics characterization of the compound manipulator

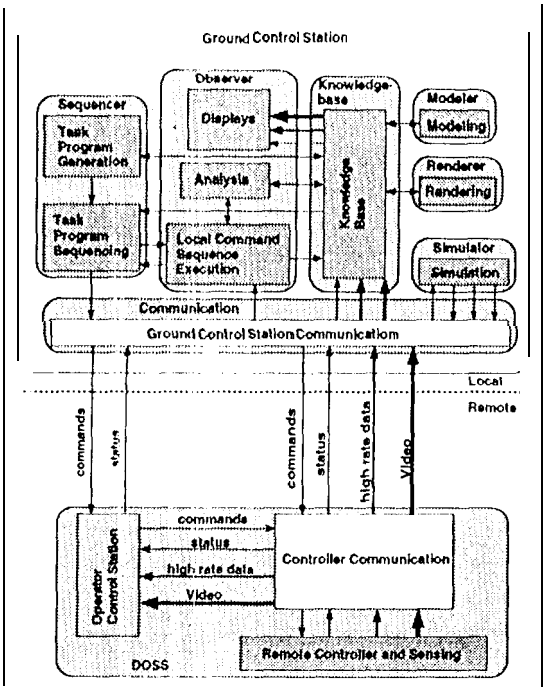

Figure 1: Ground Control Station Architecture for DOSS

system. In the SRMS-based operations stage, a subset of the fixed-base tasks will be performed for comparison with the fixed-l)aw, operation results. The stowage and deployment stage includes placing the DOSS manipulator back on the MPESS. Additionally, there may also include docking of the MPESS on the Mir space station and IVA transfer of the AFD elements to Mir. The Ground Control Station is being developed to provide the ability to command the 1)OSS system from the ground for the various stages of the flight program. The commands from the GCS may go directly to the DOSS control system or to the 1)0SS control system via the AFD Workstation.

2. The GCS Architecture

The Ground Contr o] Station architecture is shown in figure 1. The architecture is separated iuto the local site, which is the Ground Control Station, and the remote site which is the space Last'd DOSS system. The functionality of the GCS is separated into several groups, each of which could be implemented as a separate process. The Ground Control St ation is an evolution of prior ground control technology development activities [2].

The Sequencer generates and sequences task programs. This is separated into Task Program Generation and Task Program Sequencing. Task Program Generation allows the operator to create program commands by selecting commands and specifying appropriate parameters. Command parameters can be specified in the graphical simulation environment or from an appropriate list of parameters for a command. For example, for a MOVE command, the operator tells the system what frame (e.g., end effector or a grasped tool) to move to what location (e.g., a Handle of an Electronics ORU). The objective is to provide a simple way of writing individual commands with a predefined syntax. The operator specifies numeric and symbolic parameters from existing information either from a list or from interaction in the graphical environment. An interactive panel is provided for each command to aid the operator in generating a command. The Task Program Sequencing module provides both task program editing and sequencing. The operator can change, delete, and add task program commands. For example, the operator can add a new MOVE command to an existing program by first using the MOVE panel and then inserting the new command at the desired location in the task program.

Task programs are executed by sequencing each line of the task program and issuing commands both to the remote site and local site for execution. The local site execution provides the interface, displays and analysis specific to the command executing at the remote site. Task commands can be sent individually or a whole task program can be sent for execution at one time.

The Observer provides the local site interface to the operator specific to the command executing at the remote site. The Local Command Sequence Execution receives the local commands from the Sequencer and generates the displays and analysis associated with the remote site commands. The Displays are the displays for the current command and the Analysis are the local site analysis and monitoring that is done for the current command. For example, if a command requires the arm to contact an environment and thus force control is activated, this module will automatically open a window to display the force/torcluc information. Similarly, if the command is to perform an inspection task, this module will open a special inspection window to show the results. A functionality to reset system state back to a previous state is incorporated so that partially successful simulation results cau be retained aud ouly those steps needing modifications are reprogrammed and simulated.

The Knowledge Base provides information on the objects in the environment, task information such as appropriate tools for tasks on specific objects, camera characteristics, and status of tile remote site. Task programs may have symbolic data. When commands are executed, the symbolic parameters are replaced by numeric information from the Knowledge Base.

The Modeler provides modeling and calibration of the models to the real environment. The Calibration includes video calibration as discussed below. The Renderer provides rendering of a graphical representation of the remote scene. The Simulator provides a simulation of the remote site including task execution, dynamics and interaction with the remote environment.

The Communication provides communication with the remote site. Since different remote sites will have specific interface Specifications, both for communication and for command formats, this module is specific to a particular remote site. Information is translated both when sending and receiving data to interface the local and remote sites.

3. Coordinated Reactive Sequencing

It is desired to be able to send branching command sequences to the remote system for execution. These branching sequences are equivalent to state transition diagrams with commands as nodes. Since the actual sequence of commands that will be executed is not known when the branching sequence is sent to the remote site, the remote site needs to inform the local site of command results and branching. Since specific local site analysis and displays are done for specific remote site commands, the local site must have a corresponding branching sequence that is followed as informed by the remote site, This is shown iu figure 2. The local and remote sites have corresponding branching sequences, but the contents Of the specific commands are different for local and remote sites. Each local and remote site command can generate multiple subcommands and actions as implied in the figure.

4. System Implementation Status

The Ground Control Station now has various capabilities supporting task sequencing, simulation, and calibration. The target application for 1994 was a remote eddy current sensor inspection task. The various technologies were developed to a level to demonstrate

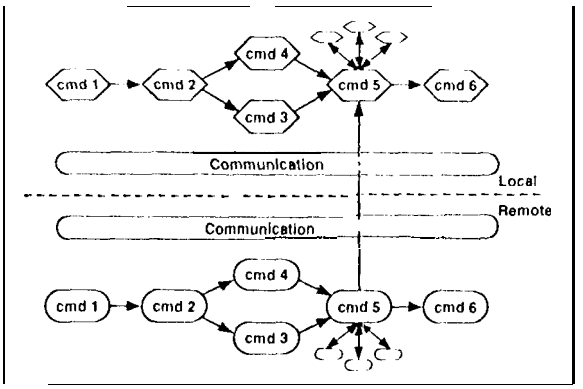

Figure 2: I,ocal-remote coordinated reactive sequencing

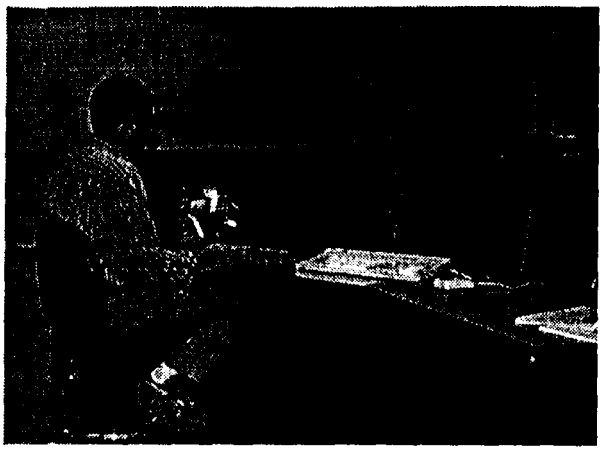

Figure 3: The ground control station

this task; the system will be further enhanced in the coming year. An operator at the GCS is shown in figure 3. The GCS operator interface screen is shown in figure 4.

4.1. Task Programming Language

A language has been developed for task programs which is called the Space Robotics Language (SRI.). SRL is an extension of the Tool Command Language (Tel) [3]. SRL is the definition of a dictionary and syntax for specifying commands to remote robots. An implementation of SRL consists of procedures for each SRI, command that may do any or all of three things: access the Knowledge Base to bind data specifications to a level appropriate for the particular remote site, translate the command into one or more remote site specific commands, and send those commands to the remote robot or simulation. The current implementation of SRL consists of procedures for commands necessary to perform an eddy current inspection task, as described below. The procedures are implemented using Tc1, therefore the translator is a Tcl interpreter that has the procedures for a particular remote site loaded.

The syntax of SRL borrows from Tel, but differs in two ways that are significant for implementation and make the commands more easily understood by an operator. First, we want a symbolic representation of data that does not require the use of the Tcl "$variable" syntax to access. Second, we want command clauses which may be comm ands themselves, but which do not require the Tcl "[command]" syntax to execute. To accomplish both requirements, SRL procedures interpret their arguments in a general fashion. First they check to see if the arguments given provide the command data in the correct numerical form. If not, they then query the Knowledge Base for the data using the arguments as symbolic identifiers. If the Knowledge Base does not have data for the specified symbol, then the procedure will return an error. If the data is found, the procedure removes the data from the argument list and then passes the remaining arguments to the interpreter as another command. Because the translator is a Tcl interpreter, the Tcl syntax forms arc still valid. Planned development will constrain task programs to (rely SRL syntax.

Once the data for a specific command has been found, the SRL procedure translates sad/or sends the command. Translating the SRL command consists of determining which remote site command or commands are necessary and mapping the SRL data specification to the form appropriate for those commands. Sending the commands consists of calling Tcl procedures that implement the interface to the remote site. The interface with the current experimental remote site [4] is a unix socket. Commands are data packets that consist of a command ID and data for the command. For each command that is defined in the remote site, a Tcl procedure is implemented in C that sends it. This Tcl-remote interface must be implemented for each remote site. The current remote site does not support sequencing of multiple commands. Therefore, sequencing is done in the GCS and individual commands are sent to the remote site.

4.2. Graphics and Video Calibration

One window on t he operator's console displays a 3-dimensional graphic representation of the remote site, including the robot arm, orbital replaceable units

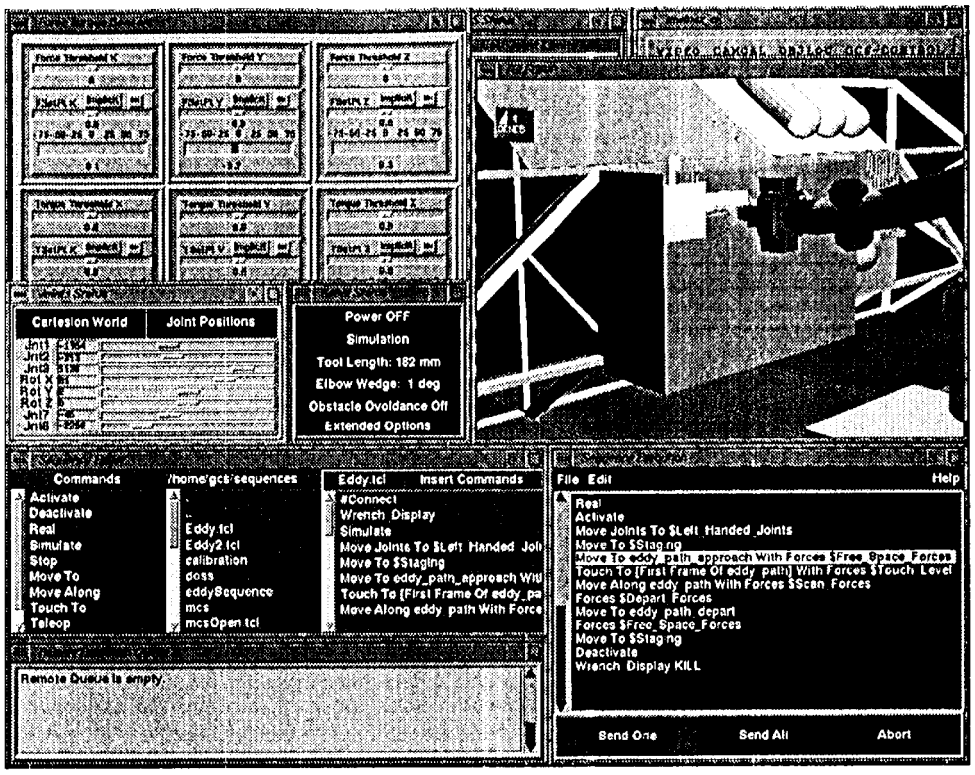

Figure 4: GCS interface for task execution

(ORUs) and Space Station truss mockup. The graphic display is driven from a hierarchical CAD model database. The arm location is kept current with periodic reports from the remote site.

Live video from any of several video cameras at the remote site is displayed on a separate monitor in the GCS console. An interactive program that commands a video switch is used to select which camera video is displayed.

Video images are also captured from any of the cameras for display on the same workstation that displays the 3-D graphic view. The actual capture takes place at the remote site, under control of commands from the GCS console, and the digitized images are transmitted back to the GCS console for display. Each frozen image is displayed in a window which is the same size as the graphics window, from which operator measurements may be made for video Calibration.

In realistic scenarios, there will be errors in the model database, either from errors in generating the database or changes since the database was generated. Effective utilization of video images returned from the remote site allows the operator to correct such errors using video calibration [5]. Video calibration consists of two distinct but similar operations: camera calibration and object localization. The main GCS display appearance is similar for both, as shown in figure 5. The video image display window appears when the operator selects video calibration mode under the "GCS-CONTROL" button of the video/graphics menu that is in the far upper right of the screen.

The camera calibration process is necessary in order to determine the correspondence between 2-dimensional image locations in a camera and 3-dimensional locations in the workspace. When a camera calibration is initiated, the GCS system guides the operator through the process of selecting 3-features for which the locations in 3-space are well-known, and measuring the corresponding 2-D image locations. The operator uses the mouse cursor to designate each feature in the 3-D graphic display window, and then uses the mouse cursor to measure the corresponding 2-D image location in the video image display window. When enough 3-D feature location to 2-D image location data sets have been collected (at least 7 sets), the operator commands the GCS system to compute a camera

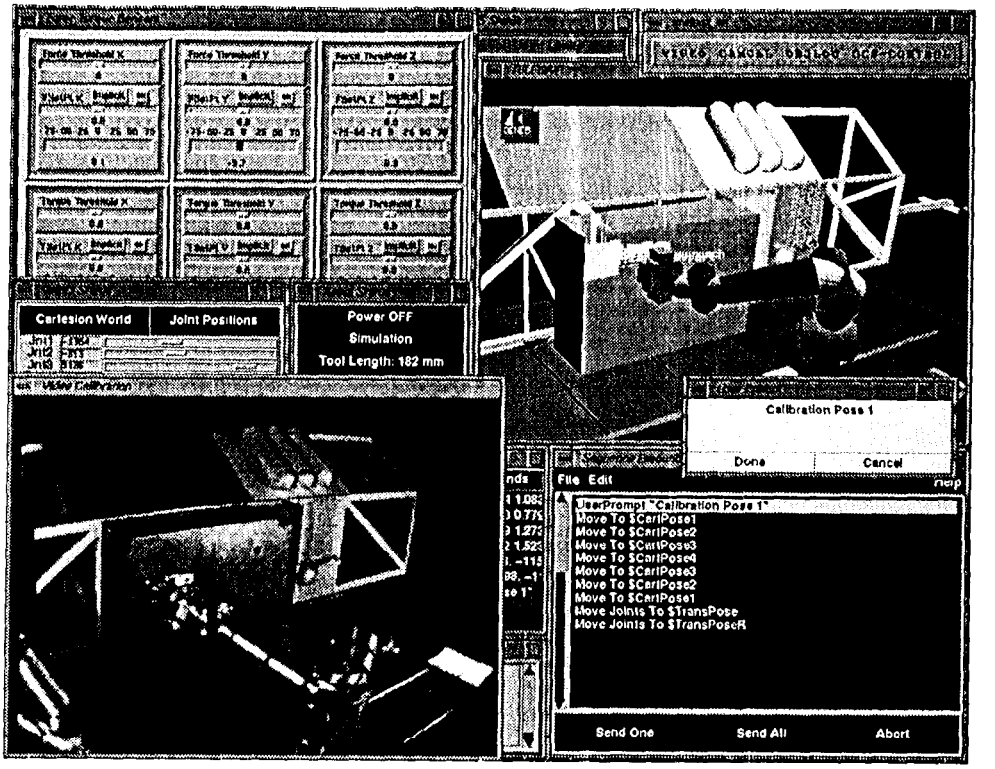

Figure 5: GCS interface for video calibration

1110(1('1.

Once camera models have been determined for at least 2 cameras, the operator may localize objects using a process similar to the camera calibration, where 3-I) features on the object of interest and corresponding 2-1) image points are selected. Data is collected from two camera locations. When sufficient points are collected, the operator commands the local ization data reduction, and a least squares solution is computed and applied to the object in the model database.

Graphical modeling and rendering was done using the TeleGRIP product from Deneb Robotics. The video calibration was developed at JPL [5] and integrated into TeleGRIP as part ofa cooperative NASA commercialization task. Therefore, the modeling and rendering, including the video calibration, are all part of one TeleGR IP process in the current system. The TeleGRIP process performs all 3-I) graphics display and video image display, and maintains almost all of the CAD model database. ,11'1, enhancements to TeleGRIP include the video image display window, all video calibration functionality, and a network interface that allows TeleGRIP to service requests from other processes, e.g., the KIio\\'ledge Base Process. The menu bar tha t appears in the far upper right of figures 5 and 4 is also part of the TeleGRIP process, and provides operator control of the JPL enhancements.

4.3. Knowledge Base I'recess

The Knowledge Base process (KB) is designed to be the access path to the TeleG R IP process, providing both update and query access to the TeleGRIP CAD model database. In addition, KB maintains an augmentati on database for data that is not representable in the TeleGRI P CAD model database, and provides a com m and channel for controlling the TeleGRIP process. KB acts as a server, responding to network requests from other processes. The KB process is implemented in Tcl with the Tcl-DP exter ision to provide network remote procedure call capability. A few JPL-added commands provide the necessary custom network communication with the TeleGRIP process.

4.4. Task Sequencing and Simulation

The Sequencer, Ol bserver and Communication modules have been combined int o one module, called

the Sequencer, in the current implementation. Simulation is done in two ways. When simulating commands for the remote site, a simulation command is sent to the remote site to put it into simulation mode. Then the commands to be simulated are sent to the remote site and executed, but instead of controlling the real manipulator, the joint commands that would have been commanded to the manipulator are returned to the local site and these are used to update the graphical simulation. When simulating 1)OSS commands, the commands are converted to equivalent IGRIP commands and sent to the TeleGRIP process directly. The GCS operator interface showing the task program for the eddy current inspection task is showu in figure 4. The Commands column provides commands that an operator can choose from when constructing or editing a task I)ro.grain. The second column provides task programs which have previously been built which the operator can choose either to execute, or to insert into another task program, or to select commands out of for insertion into another task program. The selected task program or command is displayed in the third column. The current task program is displayed in the fourth column, Sequence Execution. If the operator selects the task program name, which is displayed in the upper left of the third column, then the whole task program is inserted iuto the current task program in the fourth column. The operator can also select specific commands out of the third column and pick "Insert Commands" and these specific commands are inserted into the current task program. Commands in the current task program can also be deleted and the current task program can be saved with "save-as" and given a unique name or saved with "save". The next command to execute is highlighted aud the next command is sent by selecting the "Send One" button. For a remote site which supports receiving multiple commands, the "Send All" button would be used to send the complete sequence. The Remote Execution Queue is at the lower left of the interface and indicates what command is executing at the remote site or "Remote Queue is empty" when no command is running at the remote site, or displays an error condition.

5. Remote Execution of an Inspection Task

The task program of figure 4 performs the eddy current sensor task. This task was executed with the remote site in the JPL Remote Surface Inspection Laboratory in building 198 and the local site in the Ground Control Station laboratory in building 277. The commands of the task program are described below.

Real : Puts the remote site in real execution mode as opposed to simulate.

Activate : Activates the remote robot

Move Joints **To $Left_Handed_Joints** : Move the remote robot to specified joint angles; angles are specified as a Tcl variable.

Move To $Staging : Move to a specified frame which is specified as a Tcl variable.

Move To eddy_path_approach With Forces $Kcw.Space.Forces : Move to a specified frame; the frame is specified as a Tcl symbol "eddy_path_approach" and the procedure will see that the data is not numerical and query the knowledge base for the numerical values. The remaining arguments "With Forces $Free_Space_Forces" are executed as a separate command which sets the force control parameters to the numerical value of the Tcl variable.

Touch To [First Frame Of eddy_path] With Forces $Touch_Level_Forces : Touch a surface at a frame specified by the numerical value returned when the Tcl interpreter executes the command "First Frame of eddy_path " The SRL procedure for this command queries the knowledge base for "eddy_path".

Move Along eddy_path With Forces $Scan_Forces : Move along "eddy_path," the data for which is retrieved from the knowledge base.

Forces $Depart_Forces : Set the force control parameters to those specified by the Tcl variable.

Move To eddy-path-depart : Move to frame specified by SRL symbol "eddy_path_depart"

Forces $Free_Space_Forces : Set the force control parameters to those specified by the Tcl variable.

Move To $Staging : Move to frame specified by Tcl variable.

Deactivate : Deactivates the remote robot

Wrench_Display KILL : Closes the wrench display.

6. Simulation of the DOSS Manipulator

The DOSS manipulator was modeled in the Deneb IGRIP environment and can be commanded with SRL commands and task programs as is shown in figure 6.

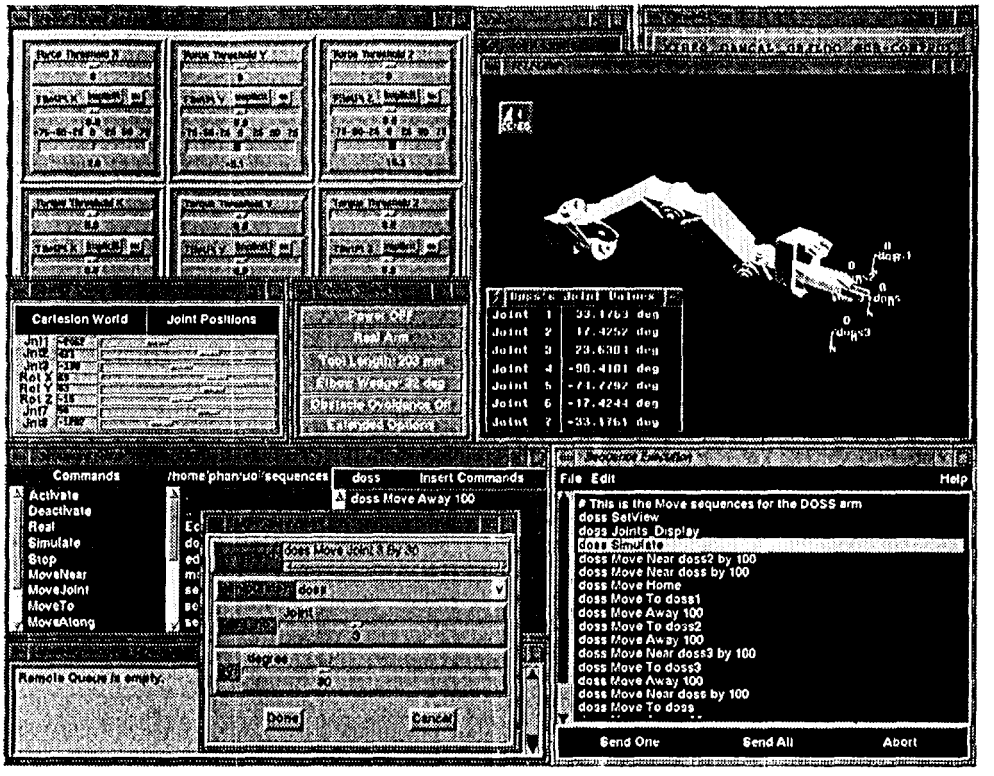

Figure6: GCS interface for 1)0SS simulation

The figure also shows a command panel which is used to generate an SRL command.

7. Conclusions

A Ground Control Station for Earth-based control of the Dexterous Orbital Servicing System (1)0SS) has been designed and an initial prototype has been developed. The system has been used for laboratory experiments using a seven degree of freedom manipulator including an eddy current inspection task at a remote location. A commanding language was developed based upon the Tool Command Language and video calibration was incorporated into the system. Also, the 1)0SS manipulator was modeled and can be commanded in simulation. A hydraulic simulator of the 1)0SS manipulator at Langley Research Center will be controlled from the GCS at J] 'I. in the coming year. The GCS system will be further enhanced in the coming year with the goal of installing the technology at Johnson Space Center for use in the 1)OSS Flight Test Program.

Acknowledgements

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] John '1', Chladek. Programmatic program requirements document for the dexterous orbiter servicing system flight test program. Tech nical Report JSC-37940 (internaldocument), Johnson Space Center, April 1994.

[2] Paul G. Backes, Joh n Beahan, and Bruce non. Interactive commandbuilding and sequencing for supervised autonomy. In *Proceedings IEEE International Conference on Robotics and Auton ation*, pages 795- 801, Atlanta, Georgia, May I 993.

[3] J. K Ousterhout. *Tcl and the Tk Tookkit.* Addison Wesley, 1994.

[4] David Lim, Thomas S. Lee, and Homayoun Seraji. A real-time control system for a mobile dexterous 7 dof arm. In *Proceedings IEEE In ternational Conference on Roboticsand Automation*, pages *1188-1195,* SanDiego, Californ ia, May 8 1? 1994.

[5] Won S. Kim. Vir t ual reality calibration for telerobotic servicing. In *Proceedings IEEE International Conference 071 Robotics and A utor nation*, volume 4, pages 2769 2775, 199 *1.*